LLNL-PROC-635098

# Structure for Storing Properties of Particles (POP)

N. R. Patel, C. M. Mattoon, B. R. Beck, N. C. Summers, D. A. Brown

April 12, 2013

International Conference on Nuclear Data for Science and Technology (ND2013)
New York, NY, United States
March 4, 2013 through March 8, 2013

# Structure for Storing Properties of Particles (PoP)

N. R. Patel,[1, *] C. M. Mattoon,[1] B. R. Beck,[1] N. C. Summers,[1] and D. A. Brown[2]
[1]*Lawrence Livermore National Laboratory, Livermore, CA 94550, USA*
[2]*National Nuclear Data Center, Brookhaven National Laboratory, Upton, NY 11973-5000, USA*
(Dated: June 4, 2013)

Evaluated nuclear databases are critical for applications such as nuclear energy, nuclear medicine, homeland security, and stockpile stewardship. Particle masses, nuclear excitation levels, and other "Properties of Particles" are essential for making evaluated nuclear databases. Currently, these properties are obtained from various databases that are stored in outdated formats. A "Properties of Particles" (PoP) structure is being designed that will allow storing all information for one or more particles in a single place, so that each evaluation, simulation, model calculation, etc. can link to the same data. Information provided in PoP will include properties of nuclei, gammas and electrons (along with other particles such as pions, as evaluations extend to higher energies). Presently, PoP includes masses from the Atomic Mass Evaluation version 2003 (AME2003) [1], and level schemes and gamma decays from the Reference Input Parameter Library (RIPL-3) [2]. The data are stored in a hierarchical structure. An example of how PoP stores nuclear masses and energy levels will be presented here.

## I. INTRODUCTION

Particle information such as masses, nuclear level energies, spins, and other "Properties of Particles" are used by various evaluations, simulations, and model calculations in nuclear physics. This information resides in various evaluated databases that are critical for applications such as nuclear energy, nuclear medicine, homeland security, and stockpile stewardship. Some of these databases are stored in outdated formats while some make it difficult for latest computer codes to access information. Furthermore, particle properties may be inconsistent between databases, giving rise to discrepancies in various evaluations. Hence, a new structure for storing "Properties of Particles" (PoP) is proposed that stores particle information in one centralized database, takes advantage of modern computing and data storage methods, defines a format that is extensible as well as human and machine-readable, includes all types of particles relevant in physics, and serves as a tool to generate an evaluated database in which all properties of particles have a defined value and follow the physics-based hierarchy.

## II. PoP STRUCTURE

In PoP each particle is endowed with all the basic information that are used to describe elementary particles such as id, symbol, mass, etc. as seen in FIG. 1. "Particle" is the fundamental storage unit in PoP. Some particles in PoP require additional information to fully describe their quantum states. For example, excited nuclear levels within an isotope are a kind of particle that have all the basic particle properties in addition to an excitation level energy (and level index). In PoP, this is handled by extending the basic particle type to store these additional data. On the other hand, a particle may share data with another particle. For example, mass of a nucleus in the ground state has to be added to the level energy to determine the mass of the nucleus in an excited state. So the mass of a nucleus is listed only for the ground state; thus, reducing redundant data and the possibility for discrepancies. PoP gives a unique *id* to each nuclear level distinguishing it from other nuclear levels. FIG. 2 shows how each excited nuclear level of an isotope will be stored. This feature will enable one to select a particular excited state as a particle.

The goal of PoP is to design a structure relevant for particles in nuclear physics. This will include properties of nuclei, photon, and electron. As evaluations in nuclear physics extend to higher energies, other particles such as pions will likely also be needed. Thus the structure of PoP must easily represent all the particles in physics. In order to represent all the particles in physics needed by evaluators, the PoP structure follows the physics-based hierarchy for organizing particle data which is similar to how the particles are classified in particle physics. FIG. 3 highlights the PoP structure designed to handle all types of particles.

PoP defines a hierarchy for storing particle informa-

* Corresponding author: infinidhi@llnl.gov

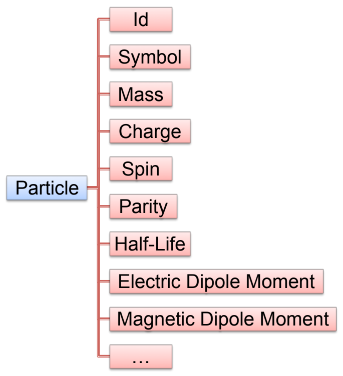

FIG. 1. PoP structure for a particle includes all the basic information used to descibe elementary particles in physics.

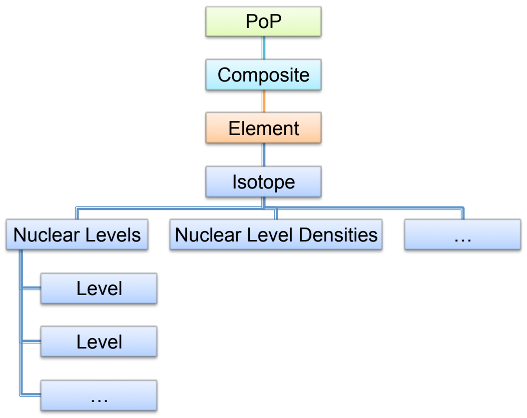

FIG. 2. PoP structure of an isotope. Level inherits from the base particle and adds properties like nuclear level energy.

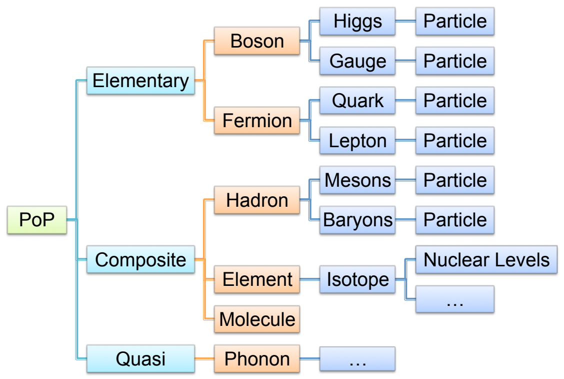

FIG. 3. The PoP structure follows the physics-based hierarchy used in particle physics.

tion. This hierarchy has already been implemented in eXtensible Markup Language (XML) and Python. The structure of PoP along with the user friendliness of XML should make it easier for nuclear model codes, evaluated data, transport codes, etc. to extract particle information from the database. XML also provides the flexibility and extensibility to accommodate new information of existing particles and new particles discovered in the future. The contents of an XML file are easy to read using any browser eliminating the need to decipher cryptic databases. The users will be able to use the PoP software on any operating systems since Python and XML are compatible to run on all of the major operating systems used today.

## III. EXAMPLES

A version of PoP has already been implemented by adopting masses from the Atomic Mass Evaluation version 2003 (AME2003) [1], and nuclear level schemes from the Reference Input Parameter Library (RIPL-3) [2]. An XML example of how PoP stores nuclear masses and energy levels is shown in FIG. 4. The ground state and the first excited state of $^{56}$Fe are stored as *level*, each with the unique *id* “Fe56_e0” and “Fe56_e1”, respectively. This modularizes the data, making it easier for users to extract just a nuclear level if that is all they need to run a simulation. If a particle decays then the tag *decayMode* specifies the decay properties for one decay. For example, “Fe56_e1” has two decay modes: “gamma” and “electronConversion”. FIG. 4 highlights how we propose to organized the particle properties such that the information is easily deciphered by humans and computer codes. FIG. 5 is an example on how to use PoP Python package to create an XML output for an electron.

## IV. DEPENDENCIES

PoP software depends on the physicalQuantityWithUncertainty (pqu) software package adopted from physicalQuantities Python module from ScientificPython[3]. Some of its features include: performing arithmetic operations of quantities with compatible units; testing whether a quantity is of energy, mass, temperature, etc. type regardless of the system of units; and storing many fundamental constants. The package supports physical quantities with uncertainty (absolute, relative, and percent), dimensionless values, and linear propagation of errors. It will allow the user to overwrite fundamental constants. With pqu, PoP will be able to provide data type to support physical quantites and checking for inconsistencies. pqu will be distributed with PoP and available for download as a standalone software package.

## V. CONCLUSION

Both PoP and pqu are still in their infancies. The structure of both are being defined and tested as we heed the needs of the nuclear physics community. In addition, PoP will come with infrastructure that will include examples on how to query the XML database, sample codes that translate the AME2003 and the RIPL-3 level schemes into an XML format, and checker codes to en-

```
<pop version="0.0.0">
  <composites>
    <elements>
      <element name="Iron" Z="26" symbol="Fe">
        <isotopes>
          <isotope id="Fe56" A="56">
            <nuclearLevels maxCompleteUpTo="Fe56_e74"
            uniqueSpinParityUpTo="Fe56_e5">
              <reference name=“RIPL-3”/>
              <level id="Fe56_e0">
                <mass name=“atomicMass”>
                  <reference name=“AME2003”/>
                  <valueIs>evaluated</valueIs>
                  <value>55934937.475e-6</value>
                  <uncertainty>0.735e-6</uncertainty>
                  <unit>amu</unit></mass>
                <energy>
                  <value>0</value>
                  <unit>MeV</unit></energy>
                <spin spinEstimationMethodFlag="u"
                originalENSDFspins="0+">0</spin>
                <parity>+</parity>
                <halfLife>stable</halfLife></level>
              <level id="Fe56_e1“>
                <energy>
                  <value>0.846776</value>
                  <unit>MeV</unit></energy>
                <spin spinEstimationMethodFlag="u"
                originalENSDFspins=“2+">2</spin>
                <parity>+</parity>
                <halfLife>
                  <value>6.07e-12</value>
                  <unit>s</unit></halfLife>
                <decayModes>
                  <decayMode name=”gamma“>
                    <energy>
                      <value>0.847</value>
                      <unit>MeV</unit></energy>
                    <decayProduct>Fe56_e0</decayProduct>
                    <decayProduct>photon</decayProduct>
                    <probability>0.9997</probability>
                  </decayMode>
                  <decayMode name=”electronConversion“>
                    <energy>
                      <value>0.847</value>
                      <unit>MeV</unit></energy>
                    <decayProduct>Fe56_e0</decayProduct>
                    <decayProduct>e-</decayProduct>
                    <probability>0.0003</probability>
                    <internalConversionCoefficient>2.943e-4
                        </internalConversionCoefficient>
                        </decayMode></decayModes></level>
                        </nuclearLevels></isotope></isotopes>
                        </element></elements></composites>
                        </pop>
```

FIG. 4. An XML example of nuclear masses and energy level information from AME2003 [1] and RIPL-3 [2] databases. Shown only the first two levels of $^{56}$Fe isotope. Users unfamiliar with XML may wish to refer to online tutorial [4]

sure the structure follows the underlying physics. Both softwares will include documentation on how to use the packages and explain the naming and terminology used. We plan to release both of these software packages and follow up with future releases taking into account the bug fixes and feedback received from the physics community.

## VI. ACKNOWLEDGEMENTS

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

```
>>> from pop import *
>>> eMass = Mass(valueIs=‘measured’,
         value=’0.510998928(11) MeV’,
         reference={
               url:’http://pdg.lbl.gov’,
               author:’J. Beringer’,
               Journal:’PRD’,
               volume:’86’,
               year:’2012’})
>>> myParticle = Particle(symbol=‘e-’, genre=‘lepton’
         mass=eMass, charge=‘-1 e’, spin=‘0.5’, parity=‘+’)
>>> print myParticle.toXML(‘ ‘ * 3)

<pop version=“0.0.0”>
  <elementary>
    <fermions>
      <leptons>
        <lepton id=“e-”>
          <charge>
            <value>-1</value>
            <unit>e</unit></charge>
          <spin>0.5</spin>
          <parity>+</parity>
          <mass>
            <reference>
              <url>http://pdg.lbl.gov</url>
              <author>J. Beringer</author>
              <journal>PRD</journal>
              <volume>86</volume>
              <year>2012</year></reference>
            <valueIs>measured</valueIs>
            <value>0.510998928</value>
            <uncertainty>1.1e-8</uncertainty>
            <unit>MeV</unit></mass></lepton></leptons>
                </fermions></elementary></pop>
```

FIG. 5. An example of the Python code and the XML output for an electron.

[1] G. Audi *et al.*, Nucl. Phys. **A729**, 337 (2003).
[2] R. Capote *et al.*, Nucl. Data Sheets **110**, 3107 (2009).
[3] K. Hinsen, “ScientificPython”, Centre National de la Recherche Scientifique, March 29, 2013, http://dirac.cnrs-orleans.fr/plone/software/scientificpython
[4] “XML Tutorial”, W3Schools, 31 May 2013, http://www.w3schools.com/xml/default.asp